**Lab 2 Report**

Name:呂聖文

Student ID:108598007

Date:2020/4/17

1. **Test Plan**
   1. **Test requirements**

The Lab 2 requires to (1) select 15 methods from 6 classes of the SUT (GeoProject), (2) design Unit test cases by using **input space partitioning (ISP)** technique for the selected methods, (3) develop test scripts to implement the test cases, (4) execute the test scripts on the selected methods, (5) report the test results, and (6) specify your experiences of designing test cases systematically using the ISP technique.

In particular, based on the statement coverage criterion, the **test requirements** for Lab 2 are to design test cases *with* ***ISP***for each selected method so that “*each statement of the method will be covered by at least one test case* and *the minimum statement coverage is 70% (greater than Lab 1)*”.

* 1. **Test Strategy**

To satisfy the test requirements listed in Section 1, a proposed strategy is to

1. select **those 10 methods that were chosen in Lab1** and **5 new methods** that are NOT selected previously. If possible, some of the methods do NOT have primitive types of input or output parameters (if possible).
2. set the objective of the minimum statement coverage to be greater than that of Lab 1 and adjust the test objective based on the time available (if necessary).
3. design the test cases for those selected methods by using the **input space partitioning (ISP)** technique.
   1. **Test activities**

To implement the proposed strategy, the following activities are planned to perform.

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Activity Name** | **Plan hours** | **Schedule Date** |
| 1 | Study GeoProject | 2 h | 4/9 |
| 2 | Learn **ISP** and JUnit | 1 day | 4/9 |
| 3 | Design test cases for the selected methods | 2days | 4/10,4/11 |
| 4 | Implement test cases | 5days | 4/12~4/16 |
| 5 | Perform tests | 1 h | 4/17 |
| *6* | Complete Lab2 report | 1days | 4/17 |

* 1. **Design Approach**

The **ISP** technique will be used to design the test cases. Specifically, the possible partitions and boundary values of input parameters shall be identified first using the **Mine Map** and **domain knowledge** (if applicable). The possible **valid** combinations of the partitions (i.e., **all combination coverage**) as well as the boundary values shall be computed for the input parameters of each selected method. Each of the partition combination can be a possible test case. *Add more test cases by considering the possible values and boundary of the outputs for the methods or by using test experiences.*

* 1. **Success criteria**

All test cases designed for the selected methods must pass and *the statement coverage should have achieved at least 70%.*

1. **Test Design**

To fulfill the test requirements listed in section 1.1, the following methods are selected and corresponding test cases are designed.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Class** | **Method** | **Test Objective** | **Inputs** | **Expected Outputs** |
|  | Base32 | encodeBase32(long i, int length) | 10進制轉32進制(i<0,length<0) | i:-75324  length:-4 | "-29jw" |
|  | Base32 | 10進制轉32進制(i<0,length=0) | i:-75324  length:0 | "-29jw" |
|  | Base32 | 10進制轉32進制(i<0,length>0) | i:-75324  length:4 | "-29jw" |
|  | Base32 | 10進制轉32進制(i=0,length<0) | i:0  length:-4 | "0" |
|  | Base32 | 10進制轉32進制(i=0,length=0) | i:0  length:0 | "0" |
|  | Base32 | 10進制轉32進制(i=0,length>0) | i:0  length:4 | "0000" |
|  | Base32 | 10進制轉32進制(i>0,length<0) | i:75324  length:-4 | "29jw |
|  | Base32 | 10進制轉32進制(i>0,length=0) | i:75324  length:0 | "29jw" |
|  | Base32 | 10進制轉32進制(i>0,length>0) | i:75324  length:4 | "29jw" |
|  | Base32 | encodeBase32(long i) | 10進制轉32進制(i<0) | i:-75324 | "0000000029jw" |
|  | Base32 | 10進制轉32進制(i=0) | i:0 | "000000000000" |
|  | Base32 | 10進制轉32進制(i>0) | i:75324 | "-0000000029jw" |
|  | Base32 | decodeBase32(String hash) | 32進制轉10進制(hash>0) | hash:"29jw" | 75324 |
|  | Base32 | 32進制轉10進制(hash=0) | hash:"0" | 0 |
|  | Base32 | 32進制轉10進制(hash<0) | hash:"-29jw" | -75324 |
|  | Base32 | getCharIndex(char ch) | 搜尋字元為陣列第幾index | ch:"b" | 10 |
|  | Base32 | 搜尋非陣列中字元是否正常拋出例外 | ch:"i" | IllegalArgumentException |
|  | Base32 | padLeftWithZerosToLength(String s,int length) | 測試補0功能  (s<0,i<0) | s:"-29jw",i:-5 | "-29jw" |
|  | Base32 | 測試補0功能  (s<0,i=0) | s:"-29jw",i:0 | "-29jw" |
|  | Base32 | 測試補0功能  (s<0,i>0) | s:"-29jw",i:-5 | "-29jw" |
|  | Base32 | 測試補0功能  (s=0,i<0) | s:"0",i:-5 | "0" |
|  | Base32 | 測試補0功能  (s=0,i=0) | s:"0",i:0 | "0" |
|  | Base32 | 測試補0功能  (s=0,i>0) | s:"0",i:5 | "00000" |
|  | Base32 | 測試補0功能  (s>0,i<0) | s:"29jw",i:-5 | "29jw" |
|  | Base32 | 測試補0功能  (s>0,i=0) | s:"29jw",i:0 | "29jw" |
|  | Base32 | 測試補0功能  (s>0,i>0) | s:"29jw",i:5 | "029jw" |
|  | CoverageLongs | getRatio() | 測試參數ratio與回傳ratio是否同一數字  (ratio<0) | ratio:-3.14 | -3.14 |
|  | CoverageLongs | 測試參數ratio與回傳ratio是否同一數字  (ratio=0) | ratio:0 | 0 |
|  | CoverageLongs | 測試參數ratio與回傳ratio是否同一數字  (ratio>0) | ratio:3.14 | 3.14 |
|  | CoverageLongs | getCount() | 回傳初始化set數量count  (count<0) | count:-3 | -3 |
|  | CoverageLongs | 回傳初始化set數量count  (count=0) | count:0 | 0 |
|  | CoverageLongs | 回傳初始化set數量count  (count>0) | count:3 | 3 |
|  | CoverageLongs | getHashLength() | 取array第一個值&0x0f後的值  (<0) | arr[0]=-10 | 6 |
|  | CoverageLongs | 取array第一個值&0x0f後的值  (=0) | arr[0]=0 | 0 |
|  | CoverageLongs | getHashLength() | 取array第一個值&0x0f後的值  (>0) | arr[0]=10 | 10 |
|  | Coverage | getHashes() | 回傳初始化之set位址  (set.size=0) | Set=[] | getHashes().size()==0 |
|  | Coverage | 回傳初始化之set位址  (set.size>0) | Set=[e1,e2,e3,e4] | getHashes().size()==4 |
|  | Coverage | getRatio() | 測試參數ratio與回傳ratio是否同一數字  (ratio<0) | ratio:-3.14 | -3.14 |
|  | Coverage | 測試參數ratio與回傳ratio是否同一數字  (ratio=0) | ratio:0 | 0 |
|  | Coverage | 測試參數ratio與回傳ratio是否同一數字  (ratio>0) | ratio:3.14 | 3.14 |
|  | Coverage | getHashLength() | 陣列第一元素長度為何  len(set[0])=0 | Set=[] | 0 |
|  | Coverage | 陣列第一元素長度為何  len(set[0])>0 | Set[0]="number1" | 7 |
|  | GeoHash | adjacentHash(String hash, Direction direction) | hash的direction方位區域為何  (邊界,top) | hash:"zz"  direction:top | "gb" |
|  | GeoHash | hash的direction方位區域為何  (邊界,bottom) | hash:"zz"  direction: bottom | "zy" |
|  | GeoHash | hash的direction方位區域為何  (邊界,left) | hash:"zz"  direction: left | "zx" |
|  | GeoHash | hash的direction方位區域為何  (邊界,right) | hash:"zz"  direction: right | "bp" |
|  | GeoHash | hash的direction方位區域為何  (非邊界,top) | hash:"ts"  direction:top | "tt" |
|  | GeoHash | hash的direction方位區域為何  (非邊界,bottom) | hash:"ts"  direction: bottom | "te" |
|  | GeoHash | hash的direction方位區域為何  (非邊界,left) | hash:"ts"  direction: left | "tk" |
|  | GeoHash | hash的direction方位區域為何  (非邊界,right) | hash:"ts"  direction: right | "tu" |
|  | GeoHash | adjacentHash(String hash, Direction direction, int steps) | hash的direction方位steps格區域為何  (邊界,top,steps<0) | hash:"zz"  direction:top  steps:-5 | "zf" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,bottom,steps<0) | hash:"zz"  direction:bottom  steps:-5 | "gu" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,left,steps<0) | hash:"zz"  direction:left  steps:-5 | "cp" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,right,steps<0) | hash:"zz"  direction:right  steps:-5 | "yx" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,top,steps=0) | hash:"zz"  direction:top  steps:0 | "zz" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,bottom,steps=0) | hash:"zz"  direction:bottom  steps:0 | "zz" |
|  | GeoHash | adjacentHash(String hash, Direction direction, int steps) | hash的direction方位steps格區域為何  (邊界,left,steps=0) | hash:"zz"  direction:left  steps:0 | "zz" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,right,steps=0) | hash:"zz"  direction:right  steps:0 | "zz" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,top,steps>0) | hash:"zz"  direction:top  steps:5 | "gu" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界, bottom,steps>0) | hash:"zz"  direction:bottom  steps:5 | "zf" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界, left,steps>0) | hash:"zz"  direction:left  steps:5 | "yx" |
|  | GeoHash | hash的direction方位steps格區域為何  (邊界,right,steps>0) | hash:"zz"  direction:right  steps:5 | "cp" |
|  | GeoHash | adjacentHash(String hash, Direction direction, int steps) | hash的direction方位steps格區域為何  (非邊界,top,steps<0) | hash:"ts"  direction:top  steps:-5 | "mx" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,bottom,steps<0) | hash: ="ts"  direction:bottom  steps:-5 | "v9" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,left,steps<0) | hash:"ts"  direction:left  steps:-5 | "wu" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,right,steps<0) | hash:"ts"  direction:right  steps:-5 | "sk" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,top,steps=0) | hash:"ts"  direction:top  steps:0 | "ts" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,bottom,steps=0) | hash:"ts"  direction:bottom  steps:0 | "ts" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,left,steps=0) | hash:"ts"  direction:left  steps:0 | "ts" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,right,steps=0) | hash:"ts"  direction:right  steps:0 | "ts" |
|  | GeoHash | hash的direction方位steps格區域為何  (非邊界,top,steps>0) | hash:"ts"  direction:top  steps:5 | "v9 |
| 1. b | GeoHash | hash的direction方位steps格區域為何  (非邊界, bottom,steps>0) | hash:"ts"  direction:bottom  steps:5 | "mx" |
| 1. b | GeoHash | hash的direction方位steps格區域為何  (非邊界, left,steps>0) | hash:"ts"  direction:left  steps:5 | "sk" |
|  | GeoHash | adjacentHash(String hash, Direction direction, int steps) | hash的direction方位steps格區域為何  (非邊界,right,steps>0) | hash:"ts"  direction:right  steps:5 | "wu" |
|  | GeoHash | decodeHash(String geohash) | Geohash轉為經緯度  (geohash<0) | Geohash:"-sb52" | getLat()==72.31201171875  getLon()==157.52197265625 |
|  | GeoHash | Geohash轉為經緯度  (geohash=0) | Geohash:"0" | getLat()==-67.5  getLon()==-157.5 |
|  | GeoHash | Geohash轉為經緯度  (geohash>0) | Geohash:"sb52" | getLat()==0.087890625  getLon()==38.49609375 |
|  | GeoHash | neighbours(String hash) | 測試w區域八方位 | (**"**w**"**) | geohash.get(0)==”t”  geohash.get(1)==”x”  geohash.get(2)==”y”  geohash.get(3)==”q”  geohash.get(4)==”v”  geohash.get(5)==”m”  geohash.get(6)==”z”  geohash.get(7)==”r” |
|  | GeoHash | right(String hash) | 回傳右邊(東方)區域geohash編號  (邊界) | hash:"zz" | "bp" |
|  | GeoHash | 回傳右邊(東方)區域geohash編號  (非邊界) | hash:"ts" | "tu" |
|  | GeoHash | left(String hash) | 回傳左邊(西方)區域geohash編號  (邊界) | hash:"zz" | "zx" |
|  | GeoHash | 回傳左邊(西方)區域geohash編號  (非邊界) | hash:"ts" | "tk" |
|  | GeoHash | bottom(String hash) | 回傳下方(南方)區域geohash編號  (邊界) | hash:"zz" | "zy" |
|  | GeoHash | 回傳下方(南方)區域geohash編號  (非邊界) | hash:"ts" | "te" |
|  | GeoHash | top(String hash) | 回傳上方(北方)區域geohash編號  (邊界) | hash:"zz" | "gb" |
|  | GeoHash | 回傳上方(北方)區域geohash編號  (非邊界) | hash:"ts" | "tt" |
|  | Geomem | add(double lat, double lon, long time, T t) | 加入一個紀錄資訊包含經緯度、時間、物件 | lat:-20  lon:150  time:5  t:"E2"(String) | getLat()==-20  getLon()==150  getTime()==5  getValue()=="E2" |
|  | Info | toString() | 顯示info資訊 | lat:3.123  lon:4.123  time:10  value:5  Option:”NTUT” | "Info [lat=3.123, lon=4.123, time=10, value=5, id=Optional.of(NTUT)]" |

The details of the design are given below:

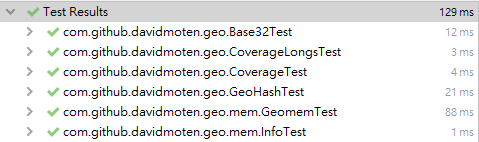
The Excel file of test cases…

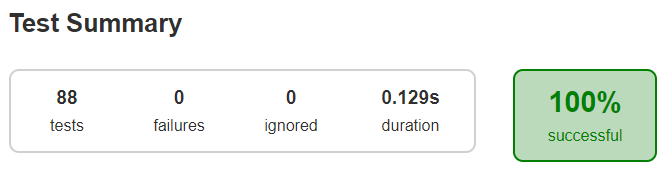
1. **Test Implementation**

The design of test cases specified in Section 2 was implemented using JUnit 4. The test scripts of 3 selected test cases are given below. The rest of the test script implementations can be found in the [link](https://github.com) (or JUnit files).

|  |  |  |
| --- | --- | --- |
| **No.** | **Test method** | **Source code** |
|  | encodeBase32\_T1() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/Base32Test.java> |
|  | encodeBase32\_T2() |
|  | encodeBase32\_T3() |
|  | encodeBase32\_T4() |
|  | encodeBase32\_T5() |
|  | encodeBase32\_T6() |
|  | encodeBase32\_T7() |
|  | encodeBase32\_T8() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/Base32Test.java> |
|  | encodeBase32\_T9() |
|  | encodeBase32\_noLength\_positive\_T1() |
|  | encodeBase32\_noLength\_zero\_T2() |
|  | encodeBase32\_noLength\_negative\_T3() |
|  | decodeBase32\_positive\_T1() |
|  | decodeBase32\_zero\_T2() |
|  | decodeBase32\_negative\_T3() |
|  | getCharIndex\_exception\_T1() |
|  | getCharIndex\_exception\_T2() |
|  | getCharIndex\_padLeftWithZerosToLength\_T1() |
|  | getCharIndex\_padLeftWithZerosToLength\_T2() |
|  | getCharIndex\_padLeftWithZerosToLength\_T3() |
|  | getCharIndex\_padLeftWithZerosToLength\_T4() |
|  | getCharIndex\_padLeftWithZerosToLength\_T5() |
|  | getCharIndex\_padLeftWithZerosToLength\_T6() |
|  | getCharIndex\_padLeftWithZerosToLength\_T7() |
|  | getCharIndex\_padLeftWithZerosToLength\_T8() |
|  | getCharIndex\_padLeftWithZerosToLength\_T9() |
|  | getRatio\_T1() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/CoverageLongsTest.java> |
|  | getRatio\_T2() |
|  | getRatio\_T3() |
|  | getCount\_T1() |
|  | getCount\_T2() |
|  | getCount\_T3() |
|  | getHashLength\_T1() |
|  | getHashLength\_T2() |
|  | getHashLength\_T3() |
|  | getHashes\_T1() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/CoverageTest.java> |
|  | getHashes\_T2() |
|  | getRatio\_T1() |
|  | getRatio\_T2() |
|  | getRatio\_T3() |
|  | getHashLength\_T1() |
|  | getHashLength\_T2() |
|  | adjacentHash\_T1() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java> |
|  | adjacentHash\_T2() |
|  | adjacentHash\_T3() |
|  | adjacentHash\_T4() |
|  | adjacentHash\_T5() |
|  | adjacentHash\_T6() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java> |
|  | adjacentHash\_T7() |
|  | adjacentHash\_T8() |
|  | adjacentHash\_steps\_T1() |
|  | adjacentHash\_steps\_T2() |
|  | adjacentHash\_steps\_T3() |
|  | adjacentHash\_steps\_T4() |
|  | adjacentHash\_steps\_T5() |
|  | adjacentHash\_steps\_T6() |
|  | adjacentHash\_steps\_T7() |
|  | adjacentHash\_steps\_T8() |
|  | adjacentHash\_steps\_T9() |
|  | adjacentHash\_steps\_T10() |
|  | adjacentHash\_steps\_T11() |
|  | adjacentHash\_steps\_T12() |
|  | adjacentHash\_steps\_T13() |
|  | adjacentHash\_steps\_T14() |
|  | adjacentHash\_steps\_T15() |
|  | adjacentHash\_steps\_T16() |
|  | adjacentHash\_steps\_T17() |
|  | adjacentHash\_steps\_T18() |
|  | adjacentHash\_steps\_T19() |
|  | adjacentHash\_steps\_T20() |
|  | adjacentHash\_steps\_T21() |
|  | adjacentHash\_steps\_T22() |
|  | adjacentHash\_steps\_T23() |
|  | adjacentHash\_steps\_T24() |
|  | decodeHash\_T1() |
|  | decodeHash\_T2() |
|  | decodeHash\_T3() |
|  | right\_T1() |
|  | right\_T2() |
|  | left\_T1() |
|  | left\_T2() |
|  | top\_T1() |
|  | top\_T2() |
|  | bottom\_T1() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java> |
|  | bottom\_T2() |
|  | neighbours() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/GeoHashTest.java> |
|  | find\_OneItemInRange() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/mem/GeomemTest.java> |
|  | testToString() | <https://stv.csie.ntut.edu.tw/108598007/GeoProject/blob/master/src/test/java/com/github/davidmoten/geo/mem/InfoTest.java> |

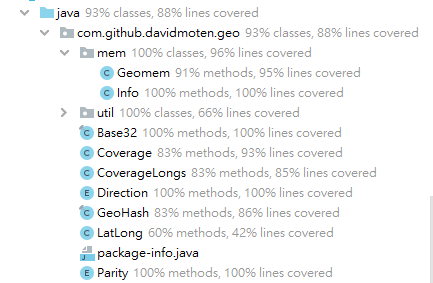
1. **Test Results**
   1. **JUnit test result snapshot**

****

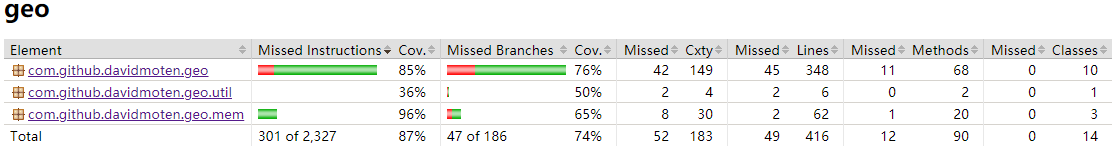
****

****

* 1. **Code coverage snapshot**
* Coverage of each selected method

****

* Total coverage

****

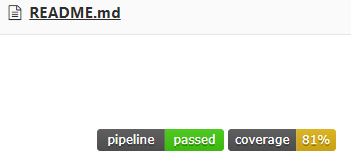
* 1. **CI result snapshot (3 iterations for CI)**
* CI#1



* CI#2



* CI#3

****

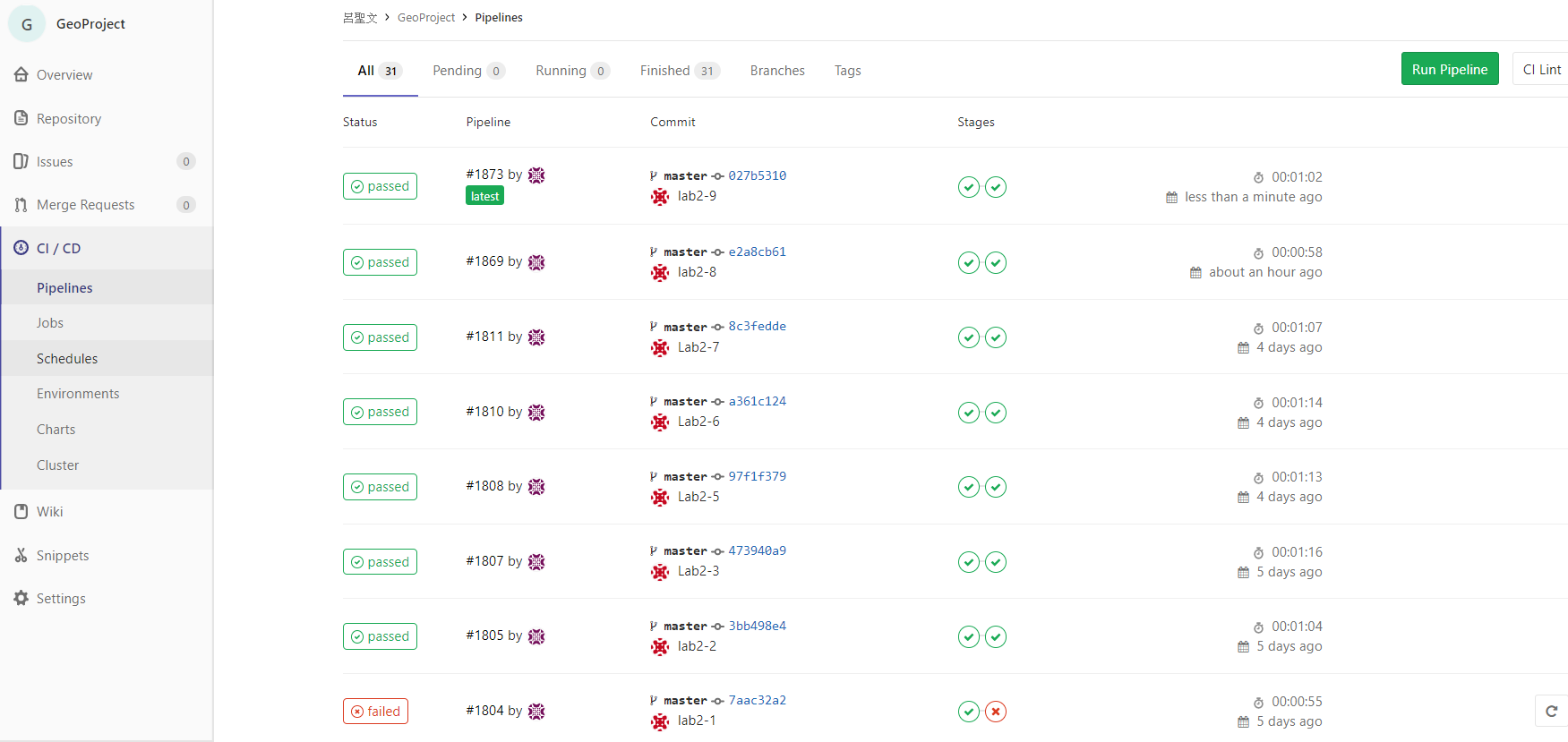
* CI#4



* CI#5

****

* CI Pipeline

****

1. **Summary**

In Lab 2, **88 test cases have been designed and implemented using JUnit and the ISP technique**. The test is conducted in 5 CI and **the execution results of the 15 test methods are all passed**. **The total statement coverage of the test is 87%.** Thus, the test requirements described in Section 1 are satisfied.

因上次作業我寫到高達84%，且內容稍亂，因此這次我重新開始撰寫，並使用ISP方式進行測試。ISP詳細內容已付在excel檔案中。

這次與Lab1最大不同為本次使用ISP的方式進行測試，使用ISP之前必須先進行規劃，規劃如何切割該方法進行測試，對於切割又是需要時間思考，怎麼切才能達到有效結果，這也是我在這次作業中花最多時間的部分，過程中也有遇到不知該如何切割才能有效測試，我自己也不是非常確定我的方式是否正確，但主要能測試該功能的運作是否正常，雖然本次作業花費非常多時間但其中也學到很多，經過本次作業我才真正了解測試的麻煩性以及需要極大的耐心才能將測試做到好。